

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

✓ W. PRICE 949/261.8433
Takakazu SHIOMI et al
NAKI-B082

5.0.09/854,268

#3

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日

Date of Application:

2000年 5月15日

出 願 番 号

Application Number:

特願2000-141492

出 願 人

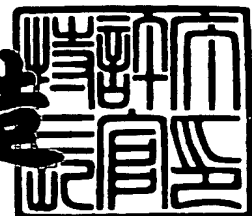
Applicant (s):

松下電器産業株式会社

2001年 1月19日

特許庁長官
Commissioner,
Patent Office

及 川 耕 造



出証番号 出証特2000-3112895

【書類名】 特許願

【整理番号】 2022520223

【提出日】 平成12年 5月15日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

【氏名】 塩見 隆一

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

【氏名】 葉山 悟

【発明者】

【住所又は居所】 広島県東広島市鏡山 3 丁目 1 0 番 1 8 号 株式会社松下電器情報システム広島研究所内

【氏名】 平本 建志

【特許出願人】

【識別番号】 000005821

【氏名又は名称】 松下電器産業株式会社

【代理人】

【識別番号】 100097445

【弁理士】

【氏名又は名称】 岩橋 文雄

【選任した代理人】

【識別番号】 100103355

【弁理士】

【氏名又は名称】 坂口 智康

【選任した代理人】

【識別番号】 100109667

【弁理士】

【氏名又は名称】 内藤 浩樹

【手数料の表示】

【予納台帳番号】 011305

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9809938

【書類名】 明細書

【発明の名称】 アプリケーション実行装置

【特許請求の範囲】

【請求項 1】 アプリケーションを記憶するアプリケーション記憶手段と、
前記アプリケーション記憶手段が記憶するアプリケーションを実行するアプリケーション実行手段と、

アプリケーションが使用する資源の回収の指示に使用される通知手段と、
前記通知手段を登録され、前記通知手段を用いて資源回収のタイミングを通知する資源回収通知手段と、

システムが備えるデバイスを制御し、アプリケーション実行手段が実行するアプリケーションに資源を提供し、前記通知手段を前記資源回収通知手段に登録し、前記資源回収通知手段より資源回収のタイミングを通知され、資源を回収するライブラリを備えることを特徴とするアプリケーション実行装置。

【請求項 2】 問い合わせに応じて前記アプリケーション実行手段が実行するアプリケーションを特定するアプリケーション特定手段を備え、

前記資源回収通知手段は、資源回収通知時に資源回収の対象であるアプリケーションを通知し、

前記ライブラリは、アプリケーションに提供する資源を前記アプリケーション特定手段によって特定されたアプリケーションと対応付けて保持し、前記資源回収通知手段から資源回収の通知を受けた際、同時に通知される資源回収対象のアプリケーションに提供した資源だけを回収することを特徴とする請求項 1 記載のアプリケーション実行装置。

【請求項 3】 前記資源回収通知手段は、資源回収通知時に資源回収の対象であるアプリケーションが複数あるとき、これら複数のアプリケーションを同時に通知することを特徴とする請求項 2 記載のアプリケーション実行装置。

【請求項 4】 アプリケーション毎に、前記資源回収通知手段を備えるアプリケーション情報管理手段と、

問い合わせに応じて前記アプリケーション実行手段が実行するアプリケーションを特定するアプリケーション特定手段を備え、

前記アプリケーション情報手段が備える前記資源回収通知手段は、前記アプリケーション情報管理手段が管理するアプリケーションから資源回収を行う場合だけ、登録されている通知手段を用いて資源回収のタイミングを通知し、

前記ライブラリは、アプリケーションに資源を提供する際、前記アプリケーション特定手段によって特定されたアプリケーションに対応するアプリケーション情報管理手段の資源回収通知に前記通知手段を登録し、前記通知手段に提供した資源を対応付け、前記資源回収通知手段から資源回収の通知を受けた際、通知に使用された通知手段に対応付けられた資源だけを回収することを特徴とする請求項 1 記載のアプリケーション実行装置。

【請求項 5】 前記資源回収通知手段に登録した前記通知手段を削除する通知手段削除手段を備えることを特徴とする請求項 1 記載のアプリケーション実行装置。

【請求項 6】 前記資源回収通知手段から資源回収対象アプリケーションとして通知されたアプリケーションの状態を通知するアプリケーション状態通知手段を備え、

前記ライブラリは、前記アプリケーション状態通知手段から得られたアプリケーションの状態に応じて資源回収を行うことを特徴とする請求項 2 記載のアプリケーション実行装置。

【請求項 7】 前記アプリケーション情報管理手段は、管理するアプリケーションの状態を通知するアプリケーション状態通知手段を備え、

前記ライブラリは、前記アプリケーション状態通知手段から得られたアプリケーションの状態に応じて資源回収を行うことを特徴とする請求項 4 記載のアプリケーション実行装置。

【請求項 8】 コンピュータ読み取り可能な記録媒体であって、アプリケーションを記憶するアプリケーション記憶手段を備える装置において、

前記アプリケーション記憶手段が記憶するアプリケーションを実行するアプリケーション実行手段と、

アプリケーションが使用する資源を回収すべきタイミングを通知手段と、

前記通知手段を登録され、前記通知手段を用いて資源回収のタイミングを通知する資源回収通知手段と、

システムが備えるデバイスを制御し、アプリケーション実行手段が実行するアプリケーションに資源を提供し、前記通知手段を前記資源回収通知手段に登録し、前記資源回収通知手段より資源回収のタイミングを通知され、資源を回収するライブラリの各機能を発揮するプログラムを記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、任意のアプリケーションを実行するアプリケーション実行装置において、アプリケーションが終了時に解放しなかった資源を回収する機構に関する。

【 0 0 0 2 】

【従来の技術】

従来のアプリケーション実行装置における資源回収方法は、特開平 8 - 1 2 3 7 0 0 「資源管理方法」に記述されている。アプリケーションはタスクに割り当てられ、タスクは複数のスレッドを有しアプリケーションを実行する。アプリケーションが使用する資源は、スレッドに関するものはスレッド単位で管理され、タスクに関する物はタスク単位で管理され、システム全体に関する物は、システム全体で管理される。図 1 2 は、これら管理状態を図示したものである。空間 1 2 0 1 はシステム全体を表し、システム全体に関する資源は空間資源 1 2 0 4 で管理される。システム中には複数のタスク 1 2 0 2 が存在し、タスクに関する資源は、タスク資源 1 2 0 5 で管理される。タスク 1 2 0 2 は複数のスレッド 1 2 0 3 を有し、スレッドに関する資源はスレッド資源 1 2 0 6 で管理される。管理されている資源は、スレッド終了時にはスレッドに関する資源だけが回収され、タスク終了時にはタスクに関する資源とその中に含まれるスレッドに関する資源が回収される。

【 0 0 0 3 】

【発明が解決しようとする課題】

しかしながら、従来の資源回収方式では、システムが使用する資源に関して、いつ回収すべきものなのか、どのような手順で回収すべきなのかを知らないなければならない。このため、システムに新規のデバイスなどを追加する場合、資源を管理する方法をシステムに組み込むため、システム自体の修正・改善・追加をしなければならない。本発明は、追加される資源の回収時期や回収方法をシステム本体に追加する必要がない、資源回収機能を持つ、アプリケーション実行装置を提供する。

【0 0 0 4】

【課題を解決するための手段】

上記課題を解決するため本発明は、アプリケーションを記憶するアプリケーション記憶手段と、前記アプリケーション記憶手段が記憶するアプリケーションを実行するアプリケーション実行手段と、アプリケーションが使用する資源の回収の指示に使用される通知手段と、前記通知手段を登録され、前記通知手段を用いて資源回収のタイミングを通知する資源回収通知手段と、システムが備えるデバイスを制御し、アプリケーション実行手段が実行するアプリケーションに資源を提供し、前記通知手段を前記資源回収通知手段に登録し、前記資源回収通知手段より資源回収のタイミングを通知され、資源を回収するライブラリを備えることとしている。

【0 0 0 5】

【発明の実施の形態】

以下、本発明に係るアプリケーション実行装置の具体的な実施の形態を、図面を参照しながら説明する。

【0 0 0 6】

(実施の形態 1)

図 1 は、本発明に係るアプリケーション実行装置の一般的な形態を表したものであり、アプリケーション入力部 1 0 1、アプリケーション記憶部 1 0 2、アプリケーション実行部 1 0 3、ライブラリー格納部 1 0 4、デバイス部 1 0 5、制御部 1 0 6 で構成される。

【 0 0 0 7 】

アプリケーション入力部 1 0 1 は、実行すべきアプリケーションを受け付ける。具体的にはフロッピーディスクや C D を等のメディアを通してアプリケーションが供給される場合、本構成要素はフロッピーディスクドライブ、C D ドライブ等で構成される。またネットワークや放送局からの電波をを通してアプリケーションが配信される場合は、ネットワークインターフェースボードや放送受信機器で構成される。内蔵されるアプリケーションのみを実行する装置においては、アプリケーション記憶部 1 0 2 にアプリケーションを記憶しておくだけで良いので、本要素は不要となる。

【 0 0 0 8 】

アプリケーション記憶部 1 0 2 は、実行すべきアプリケーションを記憶する。具体的には、R A M、R O M、ハードディスクなどの内蔵機器や、C D ドライブ、フロッピーディスクや、フロッピーディスクなどのリムーバブルメディア等で構成される。

【 0 0 0 9 】

アプリケーション実行部 1 0 3 は、アプリケーション記憶部 1 0 2 が記憶するアプリケーションを実行する。内部には、逐次実行部 1 0 3 a、資源回収通知部 1 0 3 b、アプリケーション識別部 1 0 3 c を構成要素として持つ。

【 0 0 1 0 】

逐次実行部 1 0 3 a はアプリケーション記憶部 1 0 2 が記憶するアプリケーションを実行する。アプリケーションがバイナリープログラムの場合、逐次実行部 1 0 3 a は、バイナリープログラムをそのまま実行する C P U 等で構成される。また、アプリケーションが J a v a バイトコードのような中間コードの場合は C P U 及びその上で実行されるインタープリタやバーチャルマシーンとよばれる中間コードを逐次解析し実行するモジュール等で構成される。アプリケーションは、システムが備えているデバイスを制御するため、デバイスを制御するためのライブラリを呼び出す。逐次実行部 1 0 3 a は、アプリケーションがライブラリを呼び出すとき、ライブラリ格納部 1 0 4 が格納するライブラリから必要なものを呼び出し実行する。

【 0 0 1 1 】

資源回収通知部 1 0 3 b は、ライブラリ格納部 1 0 4 が格納するライブラリから、資源回収タイミングの通知するためのコールバック関数を受け付ける。ライブラリは規定されたインターフェースを持つコールバック関数を資源回収通知部 1 0 3 b に登録する。資源回収通知部 1 0 3 b は、資源回収を行うタイミングで登録されたコールバック関数を呼び出す。この際、資源回収対象のアプリケーションの識別子も引き渡す。

【 0 0 1 2 】

アプリケーション識別部 1 0 3 c は、ライブラリがアプリケーションから呼び出された際、呼び出したアプリケーションを特定するための識別子を提供する。ライブラリがアプリケーションから呼び出された際、アプリケーション識別部 1 0 3 c を呼び出すと、アプリケーションを特定する識別子を返してくれる。

【 0 0 1 3 】

ライブラリ格納 1 0 4 は、システムが備えるデバイスを制御するためのライブラリを格納し、具体的には ROM やハードディスクなどで構成される。ライブラリ格納部 1 0 4 は、一般に複数のライブラリ 1 0 4 a 、 1 0 4 b 、 、 1 0 4 n を格納し、各ライブラリはデバイス部 1 0 5 に属するデバイス 1 0 5 a 、 1 0 5 b 、 、 1 0 5 n をアプリケーションが制御するための関数群で構成される。また、各ライブラリは資源回収通知部 1 0 3 b に登録するためのコールバック関数を備え、ライブラリが制御するデバイスがアプリケーションによって確保され使用されている状態を保持する。各ライブラリはアプリケーションによってデバイスを確保し制御する際に、コールバック関数を資源回収通知部 1 0 3 b に登録する。また、アプリケーション識別部 1 0 3 c を呼び出し、アプリケーション識別子を獲得し、確保した資源と組にして保持しておく。

【 0 0 1 4 】

アプリケーションが異常終了などにより、ライブラリを通して確保した資源を解放しなかった場合、予め登録したコールバック関数が、資源回収通知部 1 0 3 b より資源回収のタイミングで呼び出される。この時に、アプリケーション識別子が渡されるので、保持しているアプリケーション識別子と資源との組を参照し

、渡されたアプリケーション識別子に対応する資源が解放されてなければ、この資源を回収し、他のアプリケーションが使用できるようにする。

【 0 0 1 5 】

ファイルシステムを制御するファイルシステムライブラリが存在する場合を想定し、具体的に説明する。図 2 は、ファイルシステムライブラリが保持するアプリケーション識別子と資源を組にした情報を保持するテーブルデータである。（1）は、アプリケーション識別子「1」を持つアプリケーションがファイル「a. txt」を使用している状態を表す。この時、既にファイルシステムライブラリはコールバック関数を資源回収通知部 1 0 3 b に登録している。

【 0 0 1 6 】

新しく「b. txt」を使用するためファイルシステムライブラリの関数が呼び出されたとする。ファイルシステムライブラリはコールバック関数は既に登録しているので、再度登録することを行わない。ファイルシステムライブラリはアプリケーション識別部 1 0 3 c を呼び出し、アプリケーション識別子「2」を得る。ファイルシステムライブラリは、ファイル「b. txt」と獲得したアプリケーション識別子を組として保持しておく。その結果、テーブルデータは図 2（2）のように更新される。このあと、確保したファイルをアプリケーションが制御できるようにする。

【 0 0 1 7 】

アプリケーション識別子「2」を持つアプリケーションが異常終了し、ファイル資源を解放せずに終了したとする。資源回収通知部 1 0 3 b は、アプリケーション識別子「2」が確保した資源を回収するために、ファイルシステムライブラリが登録したコールバック関数を呼び出す。ファイルシステムライブラリは図 2（2）の状態のテーブルデータを参照し、ファイル「b. txt」を解放し、テーブルデータを図 2（1）の状態に戻す。

【 0 0 1 8 】

デバイス部 1 0 5 はシステムが備える全デバイスの集合である。具体的には、メモリーデバイス、ディスプレイデバイス、入力デバイス（キーボード、リモコン等）、ファイルシステム、ネットワークデバイスなどが含まれ、それぞれのデ

バイスはライブラリ格納部 1 0 6 が格納するライブラリによってアプリケーションから制御可能である。

【 0 0 1 9 】

制御部 1 0 6 は、システム上で動作するアプリケーションの開始、停止、再開、終了などを指示する。

【 0 0 2 0 】

アプリケーションの開始は、アプリケーション入力部 1 0 1 に、アプリケーションを受け付け、アプリケーション記憶部 1 0 2 に記憶する指示を行い、その後、逐次実行部 1 0 3 a にアプリケーションを実行する指示を出す。

【 0 0 2 1 】

アプリケーションの停止は、逐次実行部 1 0 3 a に実行しているアプリケーションの中断を指示する。また、場合によっては資源回収通知部 1 0 3 b に中断したアプリケーションの資源回収を指示する。この際、資源回収通知部 1 0 3 b は登録されたコールバック関数を呼び出し、停止されたアプリケーションのアプリケーション識別子を引き渡す。

【 0 0 2 2 】

アプリケーションの再開では、逐次実行部 1 0 3 a に停止しているアプリケーションを実行する指示を出す。

【 0 0 2 3 】

アプリケーションの終了は、逐次実行部 1 0 3 a に実行しているアプリケーションの終了を指示する。また、資源回収通知部 1 0 3 b に終了したアプリケーションの資源回収を指示する。この際、資源回収通知部 1 0 3 b は登録されたコールバック関数を呼び出し、終了されたアプリケーションのアプリケーション識別子を引き渡す。

【 0 0 2 4 】

図 3 は、本実施の形態のアプリケーション実行装置において、ライブラリが呼び出され資源が獲得される動作を説明するフローチャートである。

【 0 0 2 5 】

ライブラリがアプリケーションから資源を確保する指示を受けると、まず要求

された資源を確保する（ステップ S 3 0 1）。次に、コールバック関数を既に登録しているかどうかを判断し（ステップ S 3 0 2）、登録していなければ資源回収通知部 1 0 3 b にコールバック関数を登録する（ステップ S 3 0 3）。アプリケーション識別部 1 0 3 c を呼び出し、アプリケーション識別子を取得し（ステップ S 3 0 4）、確保した資源とアプリケーション識別子を組として保持しておく（ステップ S 3 0 5）。これらの資源管理処理を行った後、確保した資源をアプリケーションに提供する（ステップ S 3 0 6）。

【 0 0 2 6 】

図 4 は、本実施の形態のアプリケーション実行装置において、ライブラリが登録したコールバック関数が呼び出され資源が回収される動作を説明するフローチャートである。

【 0 0 2 7 】

コールバック関数が呼び出されると、ライブラリは資源を回収するアプリケーションを特定するアプリケーション識別子を取得する（ステップ S 4 0 1）。次に、確保している資源とアプリケーション識別子の組を 1 つ取り出す（ステップ S 4 0 2）。取得したアプリケーション識別子と、確保した資源と組になっているアプリケーション識別子を比較し（ステップ S 4 0 3）、一致した場合は、アプリケーション識別子と組で保持されている資源を解放し（ステップ S 4 0 4）、確保している資源とアプリケーション識別子の組を削除する（ステップ S 4 0 5）。ステップ S 4 0 2 からステップ S 4 0 5 までを全ての確保している資源とアプリケーション識別子の組に対して行う（ステップ S 4 0 6）。

【 0 0 2 8 】

（実施の形態 2）

図 5 は、本発明に係るアプリケーション実行装置の一例として J a v a アプリケーションを実行するシステムの形態を表したものであり、アプリケーション入力部 1 0 1、アプリケーション記憶部 1 0 2、アプリケーション実行部 5 0 3、ライブラリ格納部 5 0 4、デバイス部 1 0 5、制御部 1 0 6 で構成される。第 1 の実施例と同じ構成要素は同じ符号を割り当て説明は省略する。

【 0 0 2 9 】

アプリケーション実行部 5 0 3 は、アプリケーション記憶部 1 0 2 が記憶する J a v a アプリケーションを実行する。内部には、バーチャルマシーン部 5 0 3 a、アプリケーション管理部 5 0 3 b を構成要素として持つ。

【 0 0 3 0 】

バーチャルマシーン部 5 0 3 a はアプリケーション記憶部 1 0 2 が記憶する J a v a アプリケーションのバイトコードを実行する。J a v a アプリケーションは、システムが備えているデバイスを制御するため、デバイスを制御するためのクラスライブラリを呼び出す。バーチャルマシーン部 5 0 3 a は、j a v a アプリケーションがクラスライブラリを呼び出すとき、ライブラリ格納部 5 0 4 が格納するクラスライブラリから必要なクラスを呼び出し実行する。

【 0 0 3 1 】

アプリケーション管理部 5 0 3 b は、実行されている J a v a アプリケーションを管理する。アプリケーション情報を格納するためのクラスからインスタンスを生成し、そのインスタンスにアプリケーションの情報を格納する。5 0 3 c は、アプリケーション情報を格納するインスタンスである。ここではこれをアプリケーション情報インスタンスと呼ぶ。

【 0 0 3 2 】

アプリケーション管理部 5 0 3 b は、第 1 の実施の形態におけるアプリケーション識別部 1 0 3 c の役割を備える。つまり、ライブラリがアプリケーションから呼び出された際、呼び出したアプリケーションを特定するための識別子を提供する。ここで、アプリケーション管理部 5 0 3 b は、識別子としてアプリケーション情報インスタンスを引き渡す。

【 0 0 3 3 】

アプリケーション情報インスタンスは、第 1 の実施例における資源回収通知部 1 0 3 b の役割を備える。つまり、規定されたコールバック関数を受け付け、資源回収時に登録されたコールバック関数を呼び出す。一般に j a v a 言語では、コールバック関数の登録を、コールバック関数に相当するメソッドを備えるクラスのインスタンスの登録として表現する。アプリケーション情報インスタンスに登録するクラスの形式を規定するため、J a v a 言語におけるインターフェース

を規定する。図6は、このインターフェースの一例である。このresourceCollectionListenerインターフェースは、updateメソッドを規定している。各ライブラリは、このインターフェースをimplementsし、updateメソッドに資源回収処理を記述したクラスを用意し、インスタンスを作成してアプリケーション情報インスタンスに登録する。アプリケーション情報インスタンスは、資源回収を行うタイミングで、登録されたインスタンスのupdateメソッドを呼び出す。ここで、アプリケーション情報インスタンスに登録する資源回収用のインスタンスを資源回収インスタンスと呼ぶ。

【0034】

第1の実施の形態では、ライブラリは資源回収用のコールバック関数をアプリケーションに依存せず、資源回収通知部103bに登録したが、本第2の実施の形態では、資源を使用しているアプリケーションを管理しているアプリケーション情報インスタンスに資源回収インスタンスに登録する。アプリケーション情報インスタンスは、管理しているアプリケーションから資源を回収するタイミングで資源回収インスタンスのメソッドを呼び出す。この結果、ライブラリーは使用されるアプリケーションの数だけ、資源回収インスタンスを作成し登録しなければならないが、資源回収処理は効率的である。第1の実施の形態では、一部のライブラリしか使用していないアプリケーションが停止・終了した場合にも、全てのライブラリからのコールバック関数を呼び出す必要があった。しかし、本第2の実施の形態では、資源回収インスタンスはアプリケーションと関連付けて登録されているので、停止・終了等したアプリケーションに登録された資源回収インスタンスのみが呼び出される。また、呼び出されること自体で、資源を回収すべきアプリケーションを特定できているので、資源回収インスタンスのメソッド呼び出し時にアプリケーション識別子を引き渡す必要がない。

【0035】

更に、アプリケーション情報インスタンスには、登録した資源回収インスタンスを削除するメソッドを用意しても良い。これはアプリケーションが実行中に資源を解放し、ライブラリが資源回収の通知を必要としなくなった時に、この削除

用メソッドを呼び出しアプリケーション情報インスタンスから資源回収インスタンスを削除することで、余分な資源回収インスタンスのメソッド呼び出しを減らすことが出来る。

【 0 0 3 6 】

また、回収資源は、回収するタイミングで異なるかもしれない。具体的には、アプリケーションが停止した状態で回収してもよい資源、回収してはいけない資源などがあるかもしれない。これを知るために、アプリケーション情報インスタンスにアプリケーションの状態を知るためのメソッドを用意してもよい。ライブラリーは、資源回収インスタンスのメソッドが呼び出された際、アプリケーションの状態を知るためのメソッドを呼び出し、その結果に応じて処理を行えばよい。

【 0 0 3 7 】

図 7 は、上記の機能を含むアプリケーション情報インスタンスを作るためのクラスの一例である。 `applicationProxy` は、資源回収インスタンスを受け付ける `addListener` メソッドを用意し、またこれを削除する `removeListener` メソッドも用意している。アプリケーションの状態を知るために `getStatus` メソッドを用意している。この例ではアプリケーションの情報としてアプリケーション名を保持するだけであるが、その他の情報を保持していても実施可能である。

【 0 0 3 8 】

ライブラリ格納 5 0 4 は、システムが備えるデバイスを制御するための `java` クラスライブラリを格納し、具体的には ROM やハードディスクなどで構成される。ライブラリ格納部 5 0 4 は、一般に複数のライブラリ 5 0 4 a、5 0 4 b、...、5 0 4 n を格納し、各ライブラリーはデバイス部 1 0 5 に属するデバイス 1 0 5 a、1 0 5 b、...、1 0 5 n をアプリケーションが制御するためのクラス群で構成される。また、各 `java` クラスライブラリはアプリケーション情報インスタンス 5 0 3 c に登録するための資源回収クラスを備え、`java` クラスライブラリが制御するデバイスがアプリケーションによって確保され使用されている状態を保持する。各 `java` クラスライブラリはアプリケーションによってデ

バイスを確保し制御する際に、アプリケーション管理部 1 0 3 b を呼び出し、アプリケーション情報インスタンスを取得し、資源回収インスタンスを資源回収クラスから生成し、アプリケーションに対応するアプリケーション情報インスタンス 5 0 3 c に登録する。また、確保した資源は資源回収インスタンスに対応付けて記録しておく。

【 0 0 3 9 】

アプリケーションが異常終了などにより、j a v a クラスライブラリを通して確保した資源を解放しなかった場合、予め登録した資源回収インスタンスのメソッドが、アプリケーション情報インスタンスから呼び出される。j a v a クラスライブラリは、呼び出された資源回収インスタンスに対応付けられている資源を解放する。

【 0 0 4 0 】

ファイルシステムを制御するファイルシステムクラスライブラリが存在する場合を想定し、具体的に説明する。図 8 は、ファイルシステムライブラリが用意する資源回収クラスである。資源回収クラスから生成される資源回収インスタンスは、このインスタンスが登録されるアプリケーション情報インスタンスを内部変数 a p p と対応する資源を内部変数 n a m e に保存する。メソッド u p d a t e は、資源回収時に呼び出されるメソッドである。

【 0 0 4 1 】

図 9 は、ファイルシステムクラスライブラリが保持する資源回収インスタンスを表している。9 0 1 は資源回収インスタンスを保持するインスタンステーブルである。(1) は、アプリケーション情報インスタンス（ここでは仮に「1」と表現する)「1」で管理されるアプリケーションがファイル「a. t x t」を使用している状態を表す。この時、資源回収インスタンス 9 0 2 はアプリケーション情報インスタンス「1」に既に登録されている。

【 0 0 4 2 】

新しく「b. t x t」を使用するためファイルシステムライブラリの関数が呼び出されたとする。ファイルシステムクラスライブラリは、アプリケーション管理部 5 0 4 b を呼び出し、アプリケーションに対応するアプリケーション情報イ

ンスタンス「2」を取得する。このアプリケーション情報インスタンスと既登録の資源回収インスタンス中のアプリケーション情報インスタンスを比較する。一致したものが無いので、新規の資源回収インスタンスを生成し、獲得したアプリケーション情報インスタンスに登録する。また、資源回収インスタンスには確保した資源「b. t x t」とアプリケーション情報インスタンス「2」を記録する。図9（2）この状態を表す。

【0043】

アプリケーション情報インスタンス「2」が管理するアプリケーションが異常終了し、ファイル資源を解放せずに終了したとする。アプリケーション情報インスタンス「2」は、登録された資源回収インスタンスのメソッドを呼び出す。ファイルシステムクラスライブラリは資源回収インスタンスに記録された資源、ファイル「b. t x t」を解放すし、この資源回収インスタンスも不要なので解放する。必要ならば、アプリケーション情報インスタンスの資源回収インスタンスの削除メソッドも呼び出し、再び呼び出されないようにする。

【0044】

図10は、本実施の形態のアプリケーション実行装置において、j a v a クラスライブラリが呼び出され資源が獲得される動作を説明するフローチャートである。j a v a クラスライブラリは資源を要求されると、資源を確保する（ステップS1001）。次に、アプリケーション管理部503bから資源要求してきたアプリケーションに対応するアプリケーション情報インスタンスを獲得する（ステップS1002）。j a v a クラスライブラリは、保持している資源回収インスタンスの1つを取り出し（ステップS1003）、アプリケーション管理部503bから取得したアプリケーション情報インスタンスと資源回収インスタンス中のアプリケーション情報インスタンスを比較する（ステップS1004）。一致すれば、一致したアプリケーション情報インスタンスを含む資源回収インスタンスに確保した資源を記録し（ステップS1005）、確保した資源をアプリケーションに提供する（ステップS1009）。全資源回収インスタンスに対してアプリケーション情報インスタンスの比較を行い（ステップS1006）、一致しなければ、新規に資源回収インスタンスを生成し、確保した資源と、アプリケ

ーション管理部 5 0 3 b から取得したアプリケーション情報インスタンスを記録する（ステップ S 1 0 0 7）。次に作成した資源回収インスタンスを、アプリケーション情報インスタンスに登録する（ステップ S 1 0 0 8）。そして、確保した資源をアプリケーションに提供する（ステップ S 1 0 0 9）。

【 0 0 4 5 】

図 1 1 は、本実施の形態のアプリケーション実行装置において、j a v a クラスライブラリが登録した資源回収インスタンスのメソッドが呼び出され資源が回収される動作を説明するフローチャートである。資源回収インスタンスのメソッドが呼び出されると、資源回収インスタンスに登録されているアプリケーション情報インスタンスのアプリケーション状態取得メソッドを呼び出し、アプリケーションが停止されたのか、終了されたのかなどの情報を得る（ステップ S 1 1 0 1）。その状態に応じて、資源回収すべきかどうかを判定する（ステップ S 1 1 0 2）。回収する場合は、資源回収インスタンスに登録されている資源を解放し（ステップ S 1 1 0 3）、資源回収インスタンス自体を解放する（ステップ S 1 1 0 4）。

【 0 0 4 6 】

尚、実施の形態 1 において、資源回収時にコールバック関数に通知されるアプリケーションは 1 つだけであった。しかし、複数のアプリケーションが同時に終了する場合には、同時に複数のアプリケーションの識別子をコールバック関数に通知することが可能である。ライブラリは複数のアプリケーションの終了通知を一度に受け取ることによって、より効果的な資源回収を行うことが可能な場合がありえる。例えば、あるライブラリが管理する資源が複数のアプリケーションを終了することによって全て開放される場合、管理情報の一部を修正するのではなく、管理テーブルを初期化するだけで良い場合がある。この場合、より高速に資源回収を行うことが出来る。

【 0 0 4 7 】

尚、実施の形態 2 において、アプリケーションの状態に応じて、特定の一部の資源のみを解放することも可能である。この場合、資源回収インスタンスの解放は、全資源が解放された時に行う。

【 0 0 4 8 】

【発明の効果】

以上説明したように、本発明によれば、アプリケーションを記憶するアプリケーション記憶手段と、前記アプリケーション記憶手段が記憶するアプリケーションを実行するアプリケーション実行手段と、アプリケーションが使用する資源の回収の指示に使用される通知手段と、前記通知手段を登録され、前記通知手段を用いて資源回収のタイミングを通知する資源回収通知手段と、システムが備えるデバイスを制御し、アプリケーション実行手段が実行するアプリケーションに資源を提供し、前記通知手段を前記資源回収通知手段に登録し、前記資源回収通知手段より資源回収のタイミングを通知され、資源を回収するライブラリを備えることとしている。

【 0 0 4 9 】

ライブラリは規定された通知手段を登録し、資源回収通知手段は通知手段を通じてライブラリに資源回収の指示を出すことによって、資源回収操作自体をライブラリの中に実現することができるため、システム本体の修正無しに、ライブラリの削除・登録が可能となり、システムの保守性が高まる。

【 0 0 5 0 】

更に、問い合わせに応じて前記アプリケーション実行手段が実行するアプリケーションを特定するアプリケーション特定手段を備え、前記資源回収通知手段は、資源回収通知時に資源回収の対象であるアプリケーションを通知し、前記ライブラリは、アプリケーションに提供する資源を前記アプリケーション特定手段によって特定されたアプリケーションと対応付けて保持し、前記資源回収通知手段から資源回収の通知を受けた際、同時に通知される資源回収対象のアプリケーションに提供した資源だけを回収することとしている。

【 0 0 5 1 】

複数のアプリケーションが実行されるシステムにおいて、特定のアプリケーションのみが使用する資源だけを解放することができる。

【 0 0 5 2 】

更に、前記資源回収通知手段は、資源回収通知時に資源回収の対象であるアプ

リケーションが複数あるとき、これら複数のアプリケーションを同時に通知することとしている。

【 0 0 5 3 】

ライブラリーが管理する資源によっては複数の資源を同時に解放できる場合もある。資源回収すべき複数のアプリケーションをライブラリーが同時に受け取ることにより、呼び出しによるオーバーヘッドを削減し、かつライブラリが管理する資源の特性を生かした効率的な資源回収を実現することができる。

【 0 0 5 4 】

更に、アプリケーション毎に、前記資源回収通知手段を備えるアプリケーション情報管理手段と、問い合わせに応じて前記アプリケーション実行手段が実行するアプリケーションを特定するアプリケーション特定手段を備え、前記アプリケーション情報手段が備える前記資源回収通知手段は、前記アプリケーション情報管理手段が管理するアプリケーションから資源回収を行う場合だけ、登録されている通知手段を用いて資源回収のタイミングを通知し、前記ライブラリは、アプリケーションに資源を提供する際、前記アプリケーション特定手段によって特定されたアプリケーションに対応するアプリケーション情報管理手段の資源回収通知に前記通知手段を登録し、前記通知手段に提供した資源を対応付け、前記資源回収通知手段から資源回収の通知を受けた際、通知に使用された通知手段に対応付けられた資源だけを回収する構成とすることとしている。

【 0 0 5 5 】

アプリケーションに対応してライブラリが通知手段を登録するので、資源通知手段はアプリケーションに関係するライブラリだけを呼び出すことができるので、余分な呼び出し回数を削減することが出来る。

【 0 0 5 6 】

更に、前記資源回収通知手段に登録した前記通知手段を削除する通知手段削除手段を備えることとしている。

【 0 0 5 7 】

アプリケーションが実行中に資源を解放し、ライブラリが資源回収の通知を必要としなくなった時、通知手段削除手段を用いて削除することにより、資源回収

時に余分な呼び出し回数を削減することが出来る。

【 0 0 5 8 】

更に、前記資源回収通知手段から資源回収対象アプリケーションとして通知されたアプリケーションの状態を通知するアプリケーション状態通知手段を備え、前記ライブラリは、前記アプリケーション状態通知手段から得られたアプリケーションの状態に応じて資源回収を行うこととしている。

【 0 0 5 9 】

ライブラリは、アプリケーションの状態に応じてアプリケーションが使用している資源を回収することが出来る。例えば、非表示状態のアプリケーションが使用しているグラフィックスメモリーなどを回収することなども可能である。

【 0 0 6 0 】

更に、前記アプリケーション情報管理手段は、管理するアプリケーションの状態を通知するアプリケーション状態通知手段を備え、前記ライブラリは、前記アプリケーション状態通知手段から得られたアプリケーションの状態に応じて資源回収を行う。

【 0 0 6 1 】

ライブラリは、アプリケーションの状態に応じてアプリケーションが使用している資源を回収することが出来る。例えば、非表示状態のアプリケーションが使用しているグラフィックスメモリーなどを回収することなども可能である。

【 0 0 6 2 】

更に、コンピュータ読み取り可能な記録媒体であって、アプリケーションを記憶するアプリケーション記憶手段を備える装置において、前記アプリケーション記憶手段が記憶するアプリケーションを実行するアプリケーション実行手段と、アプリケーションが使用する資源を回収すべきタイミングを通知手段と、前記通知手段を登録され、前記通知手段を用いて資源回収のタイミングを通知する資源回収通知手段と、システムが備えるデバイスを制御し、アプリケーション実行手段が実行するアプリケーションに資源を提供し、前記通知手段を前記資源回収通知手段に登録し、前記資源回収通知手段より資源回収のタイミングを通知され、資源を回収するライブラリの各機能を発揮するプログラムとすることとしている。

。このような構成によって、資源回収機構を有するアプリケーション実行装置としてコンピュータを利用することができる。

【図面の簡単な説明】

【図 1】

本発明に係る資源回収機能を有するアプリケーション実行装置の実施の形態 1 の構成図

【図 2】

(1) 実施の形態 1 に含まれるライブラリの一例であるファイルシステムライブラリが、アプリケーションとファイル資源を対応付けて保持している状態を示す図

(2) 実施の形態 1 に含まれるライブラリの一例であるファイルシステムライブラリが、アプリケーションとファイル資源を対応付けて保持している状態を示す図

【図 3】

実施の形態 1 において、資源提供の際の動作を表すフローチャート

【図 4】

実施の形態 1 において、資源回収の際の動作を表すフローチャート

【図 5】

本発明に係る資源回収機能を有するアプリケーション実行装置の実施の形態 2 の構成図

【図 6】

実施の形態 2 に含まれるアプリケーション情報インスタンスが受け付ける資源回収インスタンスを規定するインターフェースの一例を示す図

【図 7】

実施の形態 2 に含まれるアプリケーション情報インスタンスの構成の一例を示す図

【図 8】

実施の形態 2 に含まれる j a v a クラスライブラリの一例としてのファイルシステムクラスライブラリが備える資源回収インスタンスの一例を示す図

【図 9】

(1) 実施の形態 2 に含まれるライブラリの一例であるファイルシステムクラスライブラリが、資源回収インスタンスにアプリケーション情報インスタンスとファイル資源を対応付けて保持している状態を示す図

(2) 実施の形態 2 に含まれるライブラリの一例であるファイルシステムクラスライブラリが、資源回収インスタンスにアプリケーション情報インスタンスとファイル資源を対応付けて保持している状態を示す図

【図 1 0】

実施の形態 2 において、資源提供の際の動作を表すフローチャート

【図 1 1】

実施の形態 2 において、資源回収の際の動作を表すフローチャート

【図 1 2】

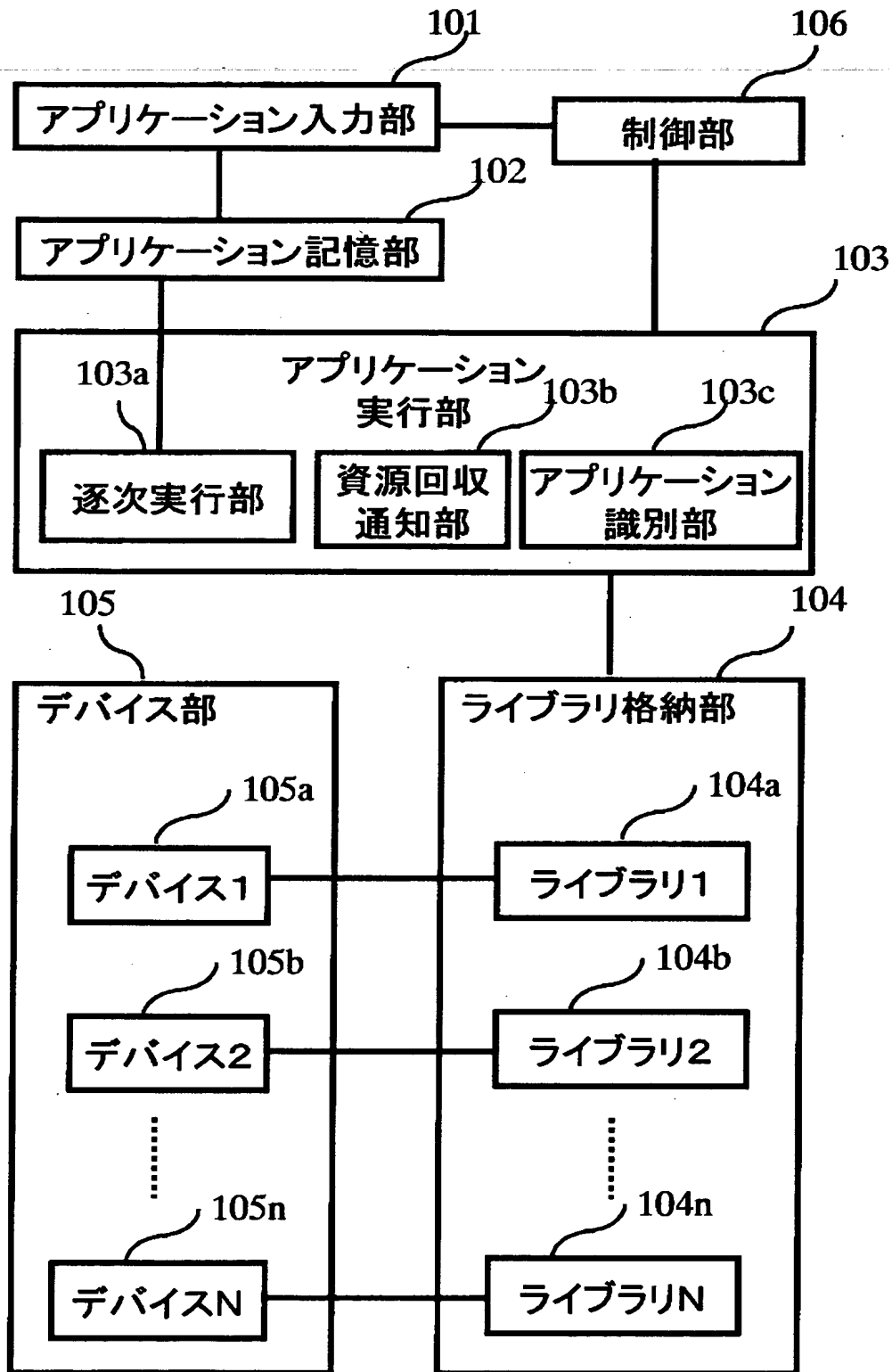
従来のアプリケーション実行装置が管理する資源の管理方法を示す図

【符号の説明】

- 1 0 1 アプリケーション入力部
- 1 0 2 アプリケーション記憶部
- 1 0 3 アプリケーション実行部
 - 1 0 3 a 逐次実行部
 - 1 0 3 b 資源回収通知部
 - 1 0 3 c アプリケーション識別部
- 1 0 4 ライブラリ格納部
- 1 0 5 デバイス部
- 1 0 6 制御部

【書類名】 図面

【図 1】



【図 2】

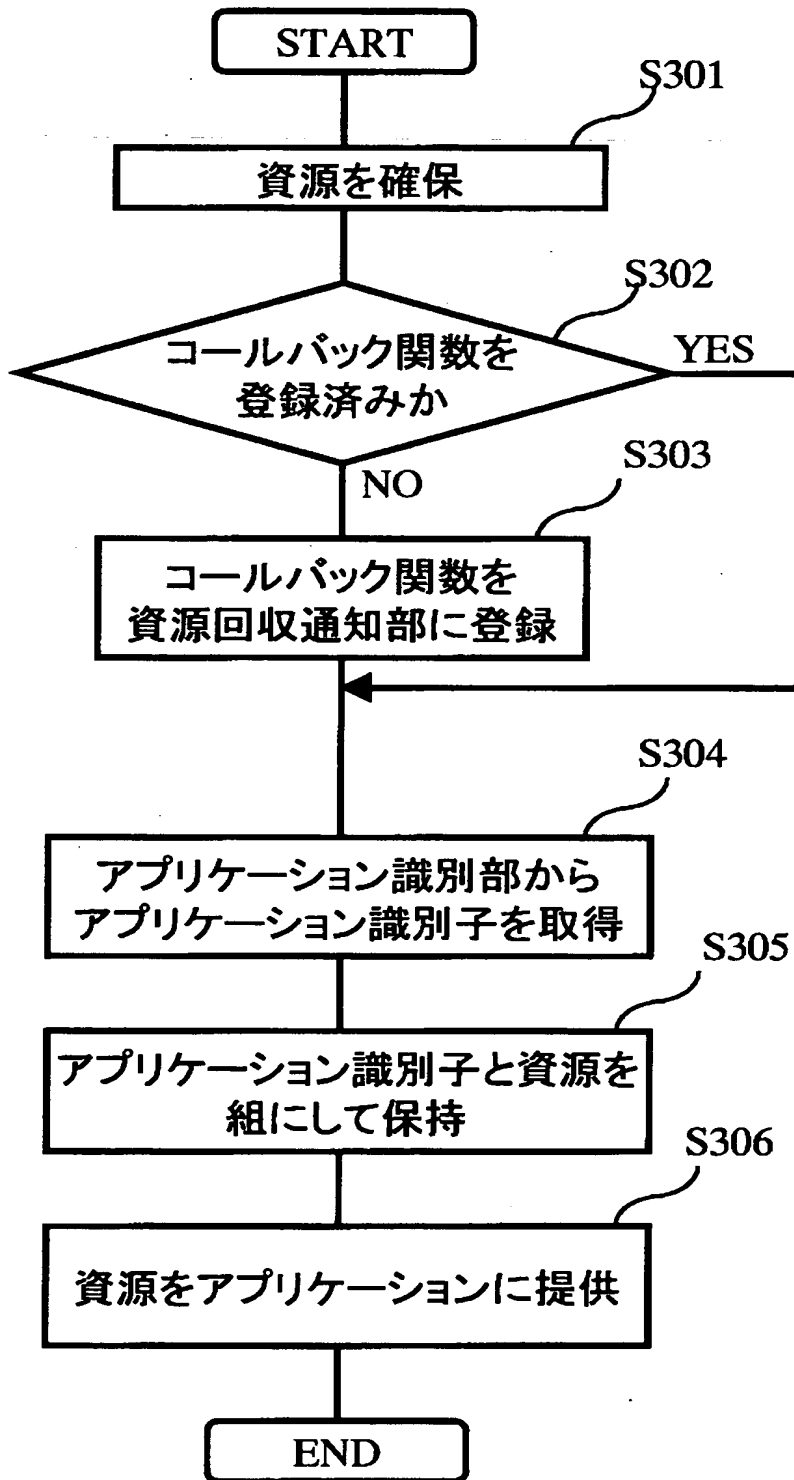
(1)

アプリケーション識別子	ファイル識別子
1	a.txt

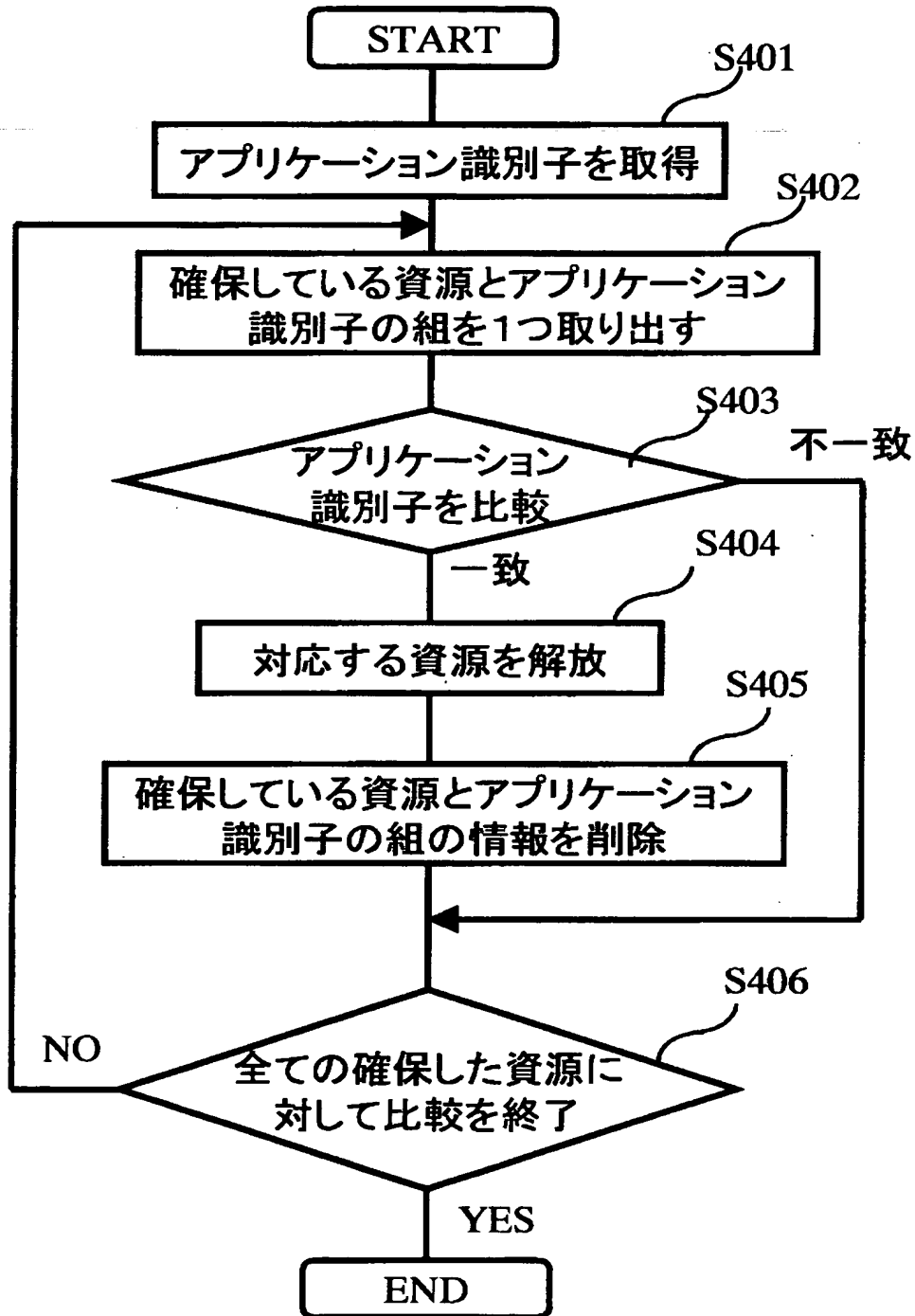
(2)

アプリケーション識別子	ファイル識別子
1	a.txt
2	b.txt

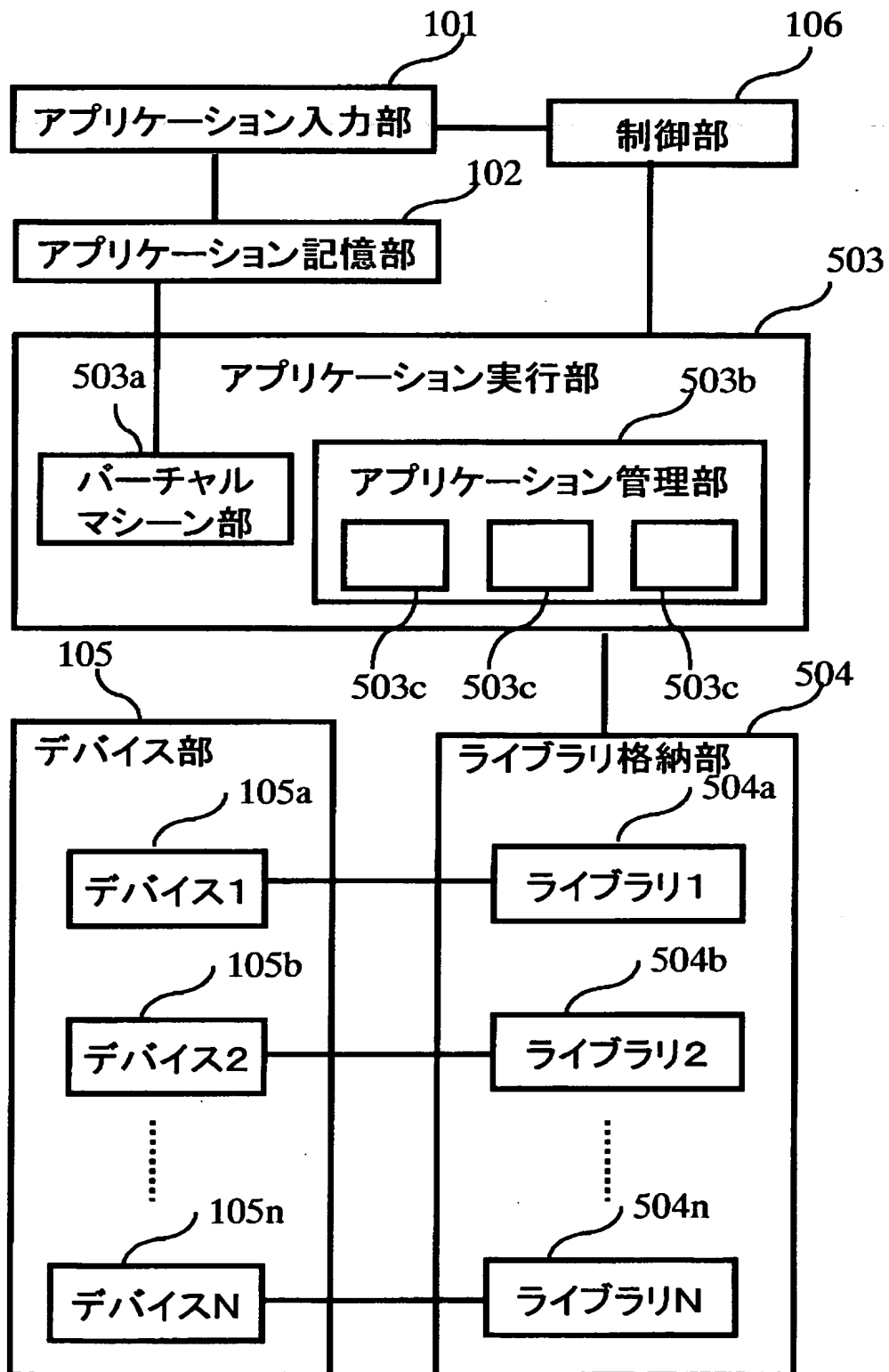
【図 3】



【図 4】



【図 5】



【図 6】

```

public interface resourceCollectionListener {
    public void update();
}

```

【図 7】

```

public class applicationProxy {
    public string name; // application name

    public int getStatus() {
        // get the status of application, pauses or destroy
    }

    public addListener(resourceCollectionListener l) {
        // register resource collection instance
    }

    public removeListener(resourceCollectionListener l) {
        // remove resource collection instance
    }
}

```

【図 8】

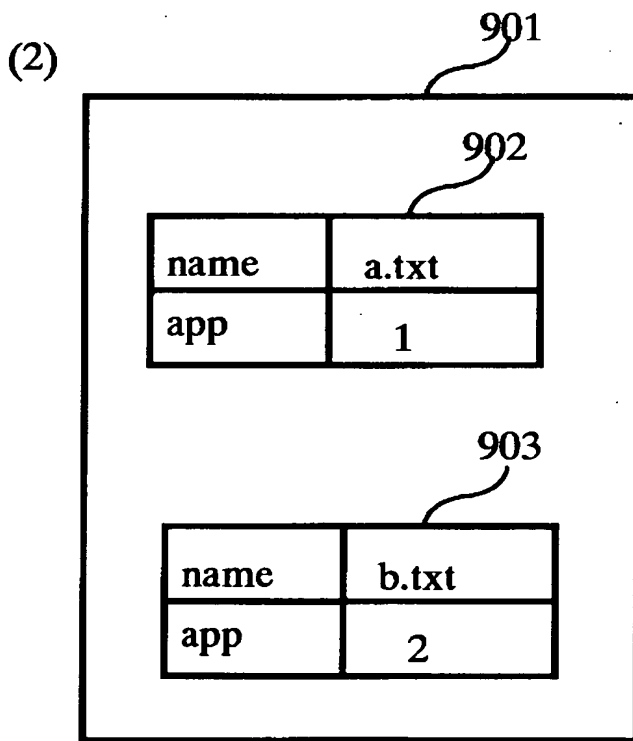
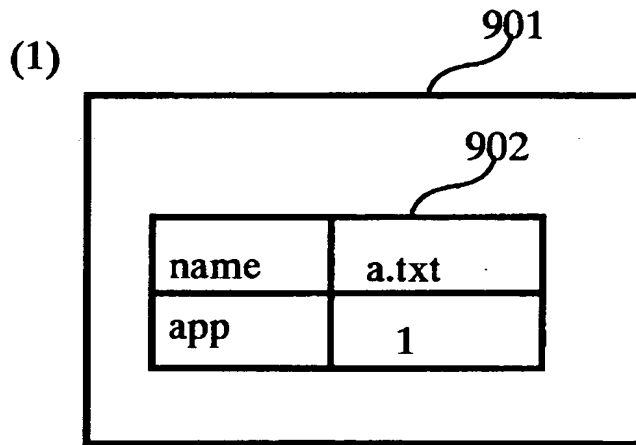
```

public class fileCollectionListener
    implements resourceCollectionListener {
    String name; // filename
    ApplicationProxy app;
        // Application information instance

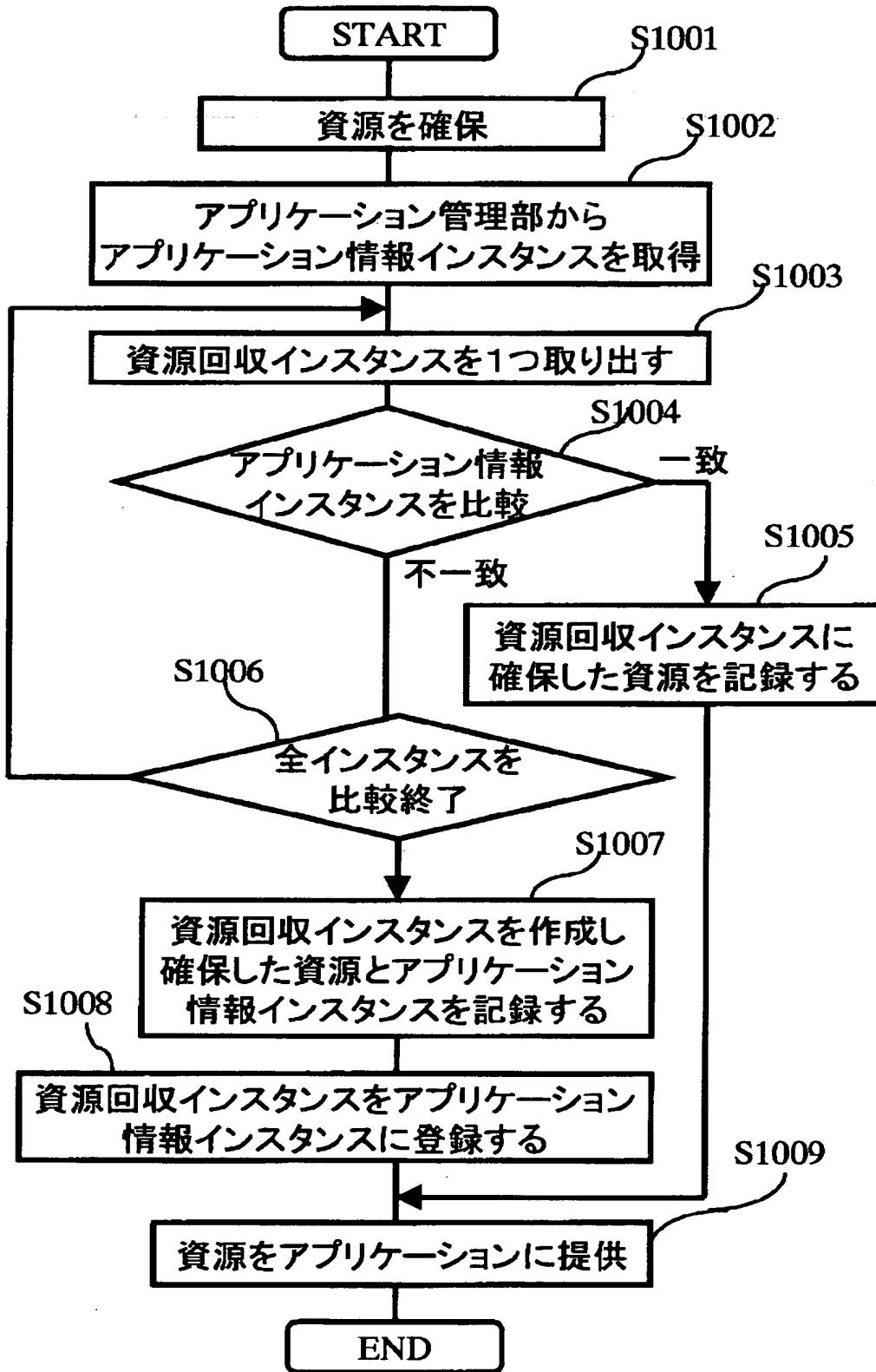
    public void update() {
        // release file which has filename
    }
}

```

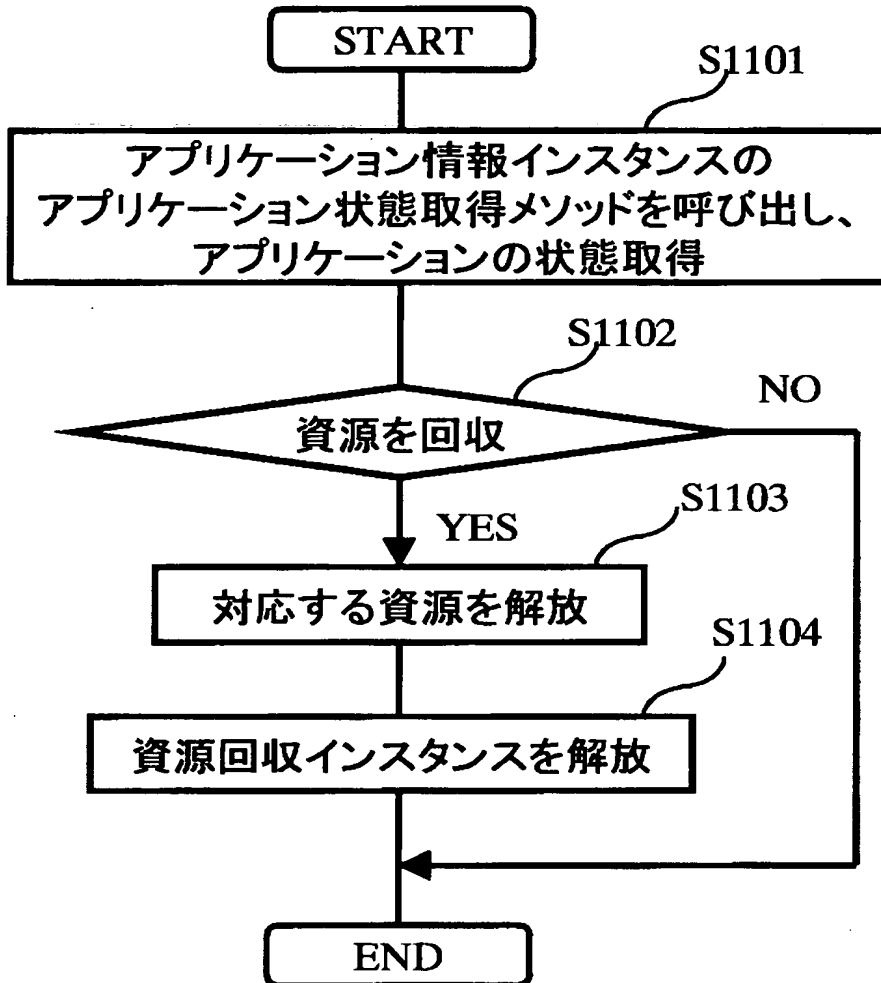
【図 9】



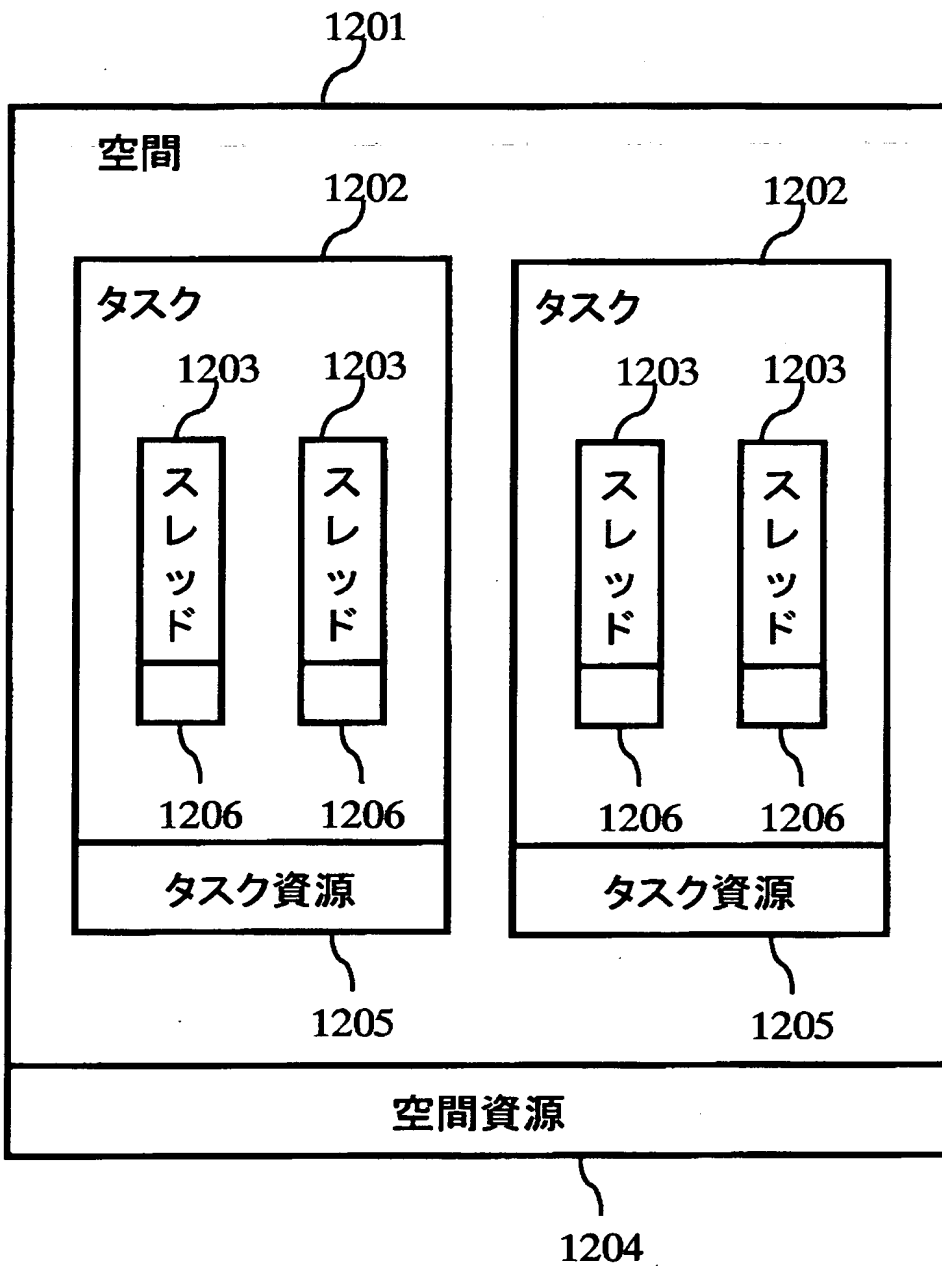
【図 1 0】



【図 1 1】



【図 1 2】



【書類名】 要約書

【要約】

【課題】 システムに新規のデバイスなどを追加する場合、資源を管理する方法をシステムに組み込むため、システム自体の修正・改善・追加をしなければならない。

【解決手段】 資源を供給するライブラリ 1 0 4 は、アプリケーションに資源を提供する際、資源回収のタイミングを通知してもらうためコールバック関数を資源回収通知部 1 0 3 b に登録する。資源回収を行うタイミングで資源回収通知手段 1 0 3 b は登録されているコールバック関数を呼び出し、ライブラリは、この呼び出しに応じて資源を回収する。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [000005821]

1. 変更年月日 1990年 8月28日
[変更理由] 新規登録
住 所 大阪府門真市大字門真1006番地
氏 名 松下電器産業株式会社